# Expérience

## Introduction

Dans cette troisième partie du travail de bachelor, je vais mener une expérience avec l’application que j’ai créée dans la deuxième partie. Cela va me permettre de tester le bon fonctionnement de cette application et aussi d’expérimenter les réseaux de neurones.

## Twitter

Nous avons à notre disposition un ensemble de tweet qui sont de nature positif ou négatif concernant des films. Notre ensemble est composé de 10662 tweets, les 5331 premiers tweets sont de polarité négative, la deuxième moitié est de polarité positive.

Dans le cas de notre expérience nous allons utiliser 90% de données pour l’entrainement et 10% pour la validation.

Un mélange des données sera effectué avant leur distribution entre l’entrainement et validation.

## Ensemble de données

Plusieurs ensembles de données ont été utilisée dans le cadre de cette expérience, nous allons les décrire dans ce chapitre.

### Twitter1

Dans twitter1 nous utilisions un vecteur one-hot, un vecteur one-hot est un vecteur unidimensionnel qui va permettre de savoir si un mot est contenu dans une phrase.

Pour générer un vecteur one-hot il faut premièrement créer un dictionnaire. Le dictionnaire va contenir tous les mots présents dans notre ensemble de données. La taille du dictionnaire va aussi représenter la taille de nos one-hot vecteur.

Pour chaque donnée nous créons un one-hot vecteur. Pour chaque mot contenu dans la donnée nous allons mettre la value à 1 dans le one-hot vecteur en fonction de la position de ce mot dans le dictionnaire.

Voici un petit exemple pour illustrer le concept :

Dictionnaire :

|  |
| --- |
| All |
| Was |
| A |
| Do |
| Making |
| Because |
| We |
| This |
| What |
| Must |
| Can |
| Good |
| Triumph |

Par conséquent la phrase « This was a triumph » va nous donner le vecteur suivant :

|  |  |
| --- | --- |
| All | 0 |
| Was | 1 |
| A | 1 |
| Do | 0 |
| Making | 0 |
| Because | 0 |
| We | 0 |
| This | 1 |
| What | 0 |
| Must | 0 |
| Can | 0 |
| Good | 0 |
| Triumph | 1 |

Du coup l’ensemble de données twitter1 sera composé de vecteur one-hot. Dans notre cas seulement les noms seront retenus. Ce qui correspond à un dictionnaire de 9210 mots différents pour 10662 données. Ce qui va nous donner 98197020 valeurs.

### Twitter2

Twitter2 reprends les mêmes principes que twitter1 avec le one-hot vecteur sauf que cette dans cet ensemble de données nous prenons en comptes les verbes en plus des noms.

Ce qui nous donne un dictionnaire avec 11000 mots donc un total de 117282000 valeurs pour les 10662 données.

### Twitter3

Dans l’ensemble de donnée twitter3 on va essayer une approche différente que le one-hot vecteur.

L’idée est de se passer de dictionnaire, à la place on va convertir les lettres en nombre grâce à la correspondance entre une lettre et le code ascii.

Par conséquent on va se retrouver avec des vecteurs de nombre fortement réduit par rapport à un vecteur one-hot.

Dans notre cas la taille du vecteur sera en fonction du tweet qui contient le nombre le plus élevé de caractères.

Par exemple, la phrase « i like it » va générer le vecteur suivant :

|  |  |
| --- | --- |
| i | 105 |
|  | 20 |
| l | 108 |
| i | 105 |
| k | 107 |
| e | 101 |
|  | 20 |
| i | 105 |
| t | 116 |

(105, 20, 108, 105, 107, 101, 20, 105, 116), Il faut encore normaliser le vecteur pour que les valeurs se trouvent entre 0 et 1. Par conséquent on utilise la valeur maximal et minimal pour redimensionner les valeurs. La formule est la suivante : (x-20) / (116-20). Ce qui nous donne le vecteur (0.885, 0, 0.916, 0.885, 0.906, 0.843, 0, 0.885, 1).

Dans l’ensemble de données de 10662 tweets, le nombre maximum de caractères est de 216. Ce qui nous donne 2302992 valeurs.

### Twitter4

Twitter4 reprend l’idée de twitter3 excepté que à la place de convertir des lettres en chiffre on va directement convertir des mots en chiffre.

Cette fois nous n’allons pas utiliser la table ascii pour faire la conversion mais nous allons utiliser la fonction native de hachage de python.

La fonction permet de retourner un nombre entier relatif.

Voici un exemple de fonctionnement avec la phrase « the cake is a lie » :

|  |  |
| --- | --- |
| The | 332310010 |
| Cake | 1867153272 |
| Is | -1500461686 |
| A | -468864544 |
| Lie | 916042657 |

On obtient le vecteur (332310010, 1867153272, -1500461686, -468864544, 916042657). Après l’utilisation de la formule de normalisation on se retrouve avec le vecteur (0.544, 1, 0, 0.306, 0.717).

Comme dans twitter1 nous utilisons seulement les noms pour les mots les autres ne seront pas pris en compte.

Du coup la phrase qui contient le plus de mot dans notre ensemble de tweets possède 52 mots. Ce qui nous donne 554424 valeurs.

### Twitter5

Exactement le même principe que twitter4 excepté qu’on ne se limite pas seulement au nom, mais on utilise tous les mots possibles. Par conséquent la phrase qui contient le plus de mot est maintenant à 62 et on se retrouve avec 661044 valeurs.

### Twitter6

### Twitter7

### Twitter8

### Twitter9

### Twitter10

### Twitter11

## Architectures

### Twitter1

### Twitter2

### Twitter3

### Twitter4

### Twitter5

### Twitter6

### Twitter7

### Twitter8

### Twitter9

### Twitter10

### Twitter11

### Twitter12

### Twitter13

### Twitter14

### Twitter15

### Twitter16

### Twitter17

## Résultats

## Analyse des résultats